Supplementary Materials: Rethinking Confidence Calibration for Failure Prediction

Fei Zhu[1,2], Zhen Cheng[1,2], Xu-Yao Zhang[1,2⋆], and Cheng-Lin Liu[1,2]

[1] NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[2] University of Chinese Academy of Sciences, Beijing, 100049, China
{zhufei2018, chengzhen2019}@ia.ac.cn, {xyz, liucl}@nlpr.ia.ac.cn

A Evaluation Metrics

A.1 Failure Prediction

AURC & E-AURC. AURC measures the area under the curve drawn by plotting the risk according to coverage. The coverage indicates the ratio of samples whose confidence estimates are higher than some confidence threshold, and the risk, also known as the selective risk [4], is an error rate computed by using those samples. A low value of AURC implies that correct and incorrect predictions can be well-separable by confidence estimates associated with samples. Inherently, AURC is affected by the predictive performance of a model. To have a unitless performance measure that can be applied across models, Geifman *et al.*, [5] introduce a normalized AURC, named Excess-AURC (E-AURC). Specifically, E-AURC can be computed by subtracting the optimal AURC, the lowest possible value for a given model, from the empirical AURC.

FPR-95%TPR. FPR-95%TPR can be interpreted as the probability that a negative (misclassified) example is predicted as a correct one when the true positive rate (TPR) is as high as 95%. True positive rate can be computed by TPR=TP/(TP+FN), where TP and FN denote the number of true positives and false negatives, respectively. The false positive rate (FPR) can be computed by FPR=FP/(FP+TN), where FP and TN denote the number of false positives and true negatives, respectively.

AUROC. AUROC measures the area under the receiver operating characteristic curve. The ROC curve depicts the relationship between true positive rate and false positive rate. This metric is a threshold-independent performance evaluation. The AUROC can be interpreted as the probability that a positive example is assigned a higher prediction score than a negative example.

AUPR-Success & AUPR-Error. AUPR is the area under the precision-recall curve. The precision-recall curve is a graph showing the precision=TP/(TP+FP)

⋆ Corresponding author.

versus recall=TP/(TP+FN). The metrics AUPR-Success and AUPR-Error indicate the area under the precision-recall curve where correct predictions and errors are specified as positives, respectively.

A.2 Confidence Calibration

ECE. Confidence calibration aims to narrow the mismatch between a model's confidence and its accuracy. As an approximation of such difference, Expected calibration error (ECE) [16] bins the predictions in $[0, 1]$ under M euqally-spaced intervals, and then averages the accuracy/confidence in each bin. Then the ECE can be computed as

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \left|\mathrm{acc}(B_m) - \mathrm{avgConf}(B_m)\right|, \tag{1}$$

where n is the number of all samples.

NLL. Negative log likelihood (NLL) is a standard measure of a probabilistic model's quality [6], which is defined as

$$\mathrm{NLL} = -\sum_{i=1}^{n} \log[\hat{p}(y_c|\boldsymbol{x_i})], \tag{2}$$

where y_c donates the element for ground-truth class. In expectation, NLL is minimized if and only if $\hat{p}(Y|\boldsymbol{X})$ recovers the truth conditional distribution.

Brier Score. Brier score [1] can be interpreted as the average mean squared error between the predicted probability and one-hot encoded label. It can be computed as

$$\mathrm{Brier} = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} [\hat{p}(y_k|\boldsymbol{x_i}) - t_k], \tag{3}$$

where $t_k = 1$ if $k = c$ (ground-truth class), and 0 otherwise.

B Details of Calibration Methods

Mixup. Mixup [22] trains a model on convex combinations of pairs of examples and their labels to encourage linear interpolating predictions. Given a pair of examples $(\boldsymbol{x}_a, \boldsymbol{y}_a)$ and $(\boldsymbol{x}_b, \boldsymbol{y}_b)$ sampled from the mini-batch, where $\boldsymbol{x}_a, \boldsymbol{x}_b$ represent different samples and $\boldsymbol{y}_a, \boldsymbol{y}_b$ denote their one-hot label vectors. Mixup applies linear interpolation to produce augmented data $(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}})$ as follows:

$$\widetilde{\boldsymbol{x}} = \lambda \boldsymbol{x}_a + (1 - \lambda)\boldsymbol{x}_b, \ \ \widetilde{\boldsymbol{y}} = \lambda \boldsymbol{y}_a + (1 - \lambda)\boldsymbol{y}_b. \tag{4}$$

The $\lambda \in [0, 1]$ is a random parameter sampled as $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$ for $\alpha \in (0, \infty)$. Thulasidasan *et al.* [19] empirically found that mixup can significantly improve

confidence calibration of DNNs. Similar calibration effect of mixup has been verified in natural language processing tasks [23].

Label Smoothing. Label Smoothing (LS) is commonly used as regularization to reduce overfitting of DNNs. Specifically, when training the model with empirical risk minimization, the one-hot label $\boldsymbol{y}$ (*i.e.* the element y_c is 1 for ground-truth class and 0 for others) is smoothed by distributing a fraction of mass over the other non ground-truth classes:

$$\widetilde{\boldsymbol{y}}_i = \begin{cases} 1-\epsilon, \text{ if } \ i=c, \\ \epsilon/(K-1), \text{ otherwise.} \end{cases} \tag{5}$$

where ϵ is a small positive constant coefficient for smoothing the one-hot label, and K is the number of training classes. Recently, Muller *et al.* [15,19] showed the favorable calibration effect of LS.

Focal Loss. Focal Loss [11] modifies the standard cross entropy loss by weighting loss components of samples in a mini-batch according to how well the model classifies them: $\mathcal{L}_f := -(1-\hat{p}_{i,y_i})^\gamma \log \hat{p}_{i,y_i}$, where γ is a strength coefficient. Intuitively, with focal loss, the gradients of correctly classified samples are restrained and those of incorrectly classified samples are emphasized. Mukhoti *et al.* [14] demonstrated that focal loss can automatically learn well-calibrated models.

CS-KD. Class-wise self-knowledge distillation (CS-KD) [21] method alleviates the overfitting problem of DNNs by penalizing the predictive distribution between the samples within the same class:

$$\mathcal{L}_{CS-KD}(\boldsymbol{x}, \boldsymbol{x'}, y, T) := \mathcal{L}_{CE}(\boldsymbol{x}, y) + \lambda_{cls} \cdot T^2 \cdot \mathrm{KL}(P(y|\boldsymbol{x'}; T) \| P(y|\boldsymbol{x}; T)), \tag{6}$$

where KL denotes the Kullback-Leibler (KL) divergence, $\mathcal{L}_{CE}$ is the standard cross-entropy loss, T is the temperature and λ_{cls} is a loss weight for the class-wise regularization. Particularly, Yun *et al.* [21] reported the positive effectiveness of CS-KD on confidence calibration.

L_p Norm. Recently, Joo *et al.* [10] explored the effect of explicit regularization strategies (*e.g.*, L_p norm in the logits space) for calibration. Specifically, the learning objective is:

$$\mathcal{L}_{L_p}(\boldsymbol{x}, y) := \mathcal{L}_{CE}(\boldsymbol{x}, y) + \lambda \|f(\boldsymbol{x})\|, \tag{7}$$

where $f(\boldsymbol{x})$ donates the logit of $\boldsymbol{x}$, and λ is a strength coefficient. Although being simple, L_p norm (*e.g.*, L_1 norm) can provide well-calibrated predictive uncertainty [10].

C Experiments

C.1 Evaluating Calibration Methods for Failure Prediction

Hyper-parameter setup of calibration methods. In our experiments, for mixup, we follow the setting in [19] to use $\alpha = 0.2$. For LS, it has been shown that

Table 1. Evaluating calibration methods for failure prediction on MobileNet and EfficientNet. AURC and E-AURC values are multiplied by 10^3 for clarity, and all remaining values are percentage.

CIFAR-10							
Network	**Method**	**AURC (↓)**	**E-AURC (↓)**	**FPR-95% TPR(↓)**	**AUROC (↑)**	**AUPR-Success(↑)**	**AUPR-Error(↑)**
MobileNet-v2	baseline [7]	**9.65±0.09**	**6.71±0.15**	**45.62±0.69**	**92.29±0.18**	**99.29±0.02**	**46.98±0.61**
	mixup [19]	10.38±0.13	8.06±0.17	47.25±0.35	90.86±0.22	99.14±0.02	42.44±0.87
	LS [15]	14.49±0.59	11.84±0.63	46.21±2.47	88.93±0.30	98.74±0.07	43.55±1.69
	Focal [14]	11.59±0.40	8.04±0.22	48.91±0.30	91.44±0.17	99.13±0.03	45.30±0.60
	CS-KD [21]	23.50±1.63	18.32±1.46	52.82±0.60	87.17±0.66	98.00±0.16	46.20±1.39
	L1 [10]	11.26±0.48	8.74±0.43	47.10±2.21	90.36±0.39	99.07±0.05	43.64±1.97
EfficientNet	baseline [7]	**16.24±0.32**	**11.24±0.45**	53.41±1.00	**90.23±0.19**	**98.78±0.05**	**47.88±1.75**
	mixup [19]	16.24±2.27	11.85±2.12	52.25±1.01	90.11±0.75	98.72±0.23	46.97±0.84
	LS [15]	24.56±0.72	19.91±0.70	51.39±0.70	87.72±0.21	97.84±0.07	47.08±0.34
	Focal [14]	18.93±0.40	13.39±0.36	55.06±1.48	89.24±0.50	98.54±0.04	47.79±1.74
	CS-KD [21]	21.15±0.90	16.77±1.03	50.93±2.09	88.05±0.74	98.18±0.11	46.57±1.79
	L1 [10]	20.21±1.04	15.62±1.28	**51.16±0.61**	88.97±0.55	98.30±0.14	48.26±1.24
CIFAR-100							
Network	**Method**	**AURC (↓)**	**E-AURC (↓)**	**FPR-95% TPR(↓)**	**AUROC (↑)**	**AUPR-Success(↑)**	**AUPR-Error(↑)**
MobileNet-v2	baseline [7]	**90.02±3.39**	**46.06±2.85**	**65.61±1.57**	**85.96±0.71**	94.05±0.37	**66.96±1.39**
	mixup [19]	93.78±0.81	46.77±1.14	67.60±1.08	85.60±0.53	**94.70±0.39**	64.44±1.00
	LS [15]	92.61±0.90	49.07±0.51	67.18±0.90	84.98±0.12	93.67±0.07	65.86±0.86
	Focal [14]	103.66±2.21	53.26±1.38	68.95±1.05	84.31±0.26	93.01±0.05	65.47±0.61
	CS-KD [21]	121.31±4.54	60.59±3.17	66.45±1.25	84.77±0.31	91.73±0.48	65.16±0.24
	L1 [10]	90.07±0.91	46.93±0.77	66.79±1.76	85.44±0.32	93.96±0.09	66.10±0.78
EfficientNet	baseline [7]	**109.88±1.58**	**55.06±0.21**	67.29±0.91	**85.18±0.24**	**92.77±0.02**	**70.95±0.78**
	mixup [19]	114.19±3.13	57.14±1.69	67.06±2.20	84.71±0.34	92.31±0.25	68.92±0.76
	LS [15]	116.48±2.63	59.73±1.38	**65.84±1.28**	84.64±0.09	91.94±0.22	69.48±0.23
	Focal [14]	132.34±2.60	65.47±3.31	69.01±1.77	83.51±0.71	90.97±0.44	69.20±0.88
	CS-KD [21]	117.75±0.86	55.60±0.80	66.97±1.16	84.90±0.44	92.61±0.09	68.23±0.65
	L1 [10]	113.47±2.77	58.56±1.28	67.32±1.32	84.33±0.32	92.16±0.20	67.87±0.54

an $\epsilon \in [0.05, 0.1]$ performs best for calibration. Therefore, $\epsilon = 0.05$ is used in our experiments. For focal loss, we set the hyperparameter $\gamma = 3$ following the suggestion in [14]. For CS-KD, we use the default setting of hyper-parameters for implementation based on their open-sourced code https://github.com/alinlab/cs-kd. For L_p norm, we use the L_1 norm, which is effective for calibration, as shown in [10], and $\lambda = 0.01$ is used in our experiments. The details of the calibration methods and definition of those hyper-parameters can be found in Section B.

Results on MobileNet and EfficientNet. We provide the results of more networks: MobileNet-v2 [8] and EfficientNet [18] in Table 1, from which we can observe similar negative effect of calibration methods on failure prediction.

Results on Tiny-ImageNet. For experiments on Tiny-ImageNet [20], the models (ResNet-18 and ResNet-50) are trained from sketch using SGD with a momentum of 0.9, an initial learning rate of 0.1, and a weight decay of 5e-4 for 90 epochs with the mini-batch size of 128. The learning rate is reduced by a factor of 10 at 40, and 70 epochs. The results are shown in Fig. 1. As can be seen, baseline has the highest AUROC values, which indicates that baseline still performs better than those compared calibration methods for failure prediction.

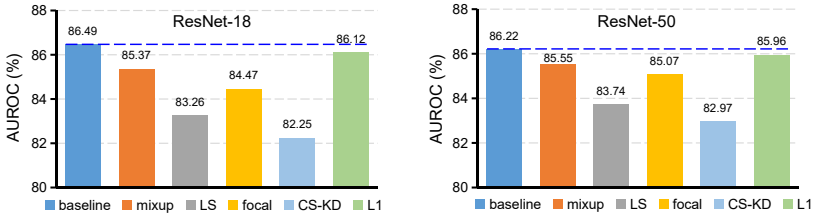

Fig. 1. Large-scale experiments on Tiny-ImageNet with ResNet-18 and ResNet-50.

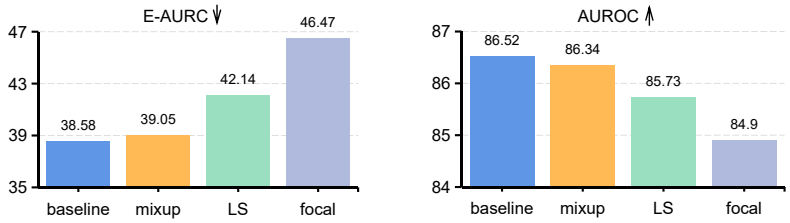

Fig. 2. Large-scale experiments on ImageNet with ResNet-50.

Results on ImageNet with ResNet-50. We perform the automatic mixed precision training using the open-sourced code https://github.com/NVIDIA/apex/tree/master/examples/imagenet. The results of on ImageNet dataset with ResNet-50 are shown in Fig. 2, from which we can observe similar negative effect of calibration methods on failure prediction.

C.2 Improving Failure Prediction by Finding Flat Minima

Pseudo-code for FMFP. Algorithm 1 gives pseudo-code for the full FMFP algorithm, which can be implemented by a few lines of codes in pytorch.

Implementation details. For CRL [13], our implementation is based on open-sourced code https://github.com/daintlab/confidence-aware-learning. For SWA and FMFP, the cyclical learning rate schedule is used as suggested in [9]. For experiments on CIFAR-10 and CIFAR-100, checkpoints at 120-th epoch of the baseline models are used as the initial point of SWA and FMFP. For experiments on Tiny-ImageNet, checkpoints at 50-th epoch of the baseline models are used as the initial point of SWA and FMFP.

Results on Tiny-ImageNet. For experiments on Tiny-ImageNet [20], the models (ResNet-18 and ResNet-50) are trained from sketch using SGD with a momentum of 0.9, an initial learning rate of 0.1, and a weight decay of 5e-4 for 90 epochs with the mini-batch size of 128. The learning rate is reduced by a factor of 10 at 40, and 70 epochs. The results are shown in Table 2. We observe that ours method (FMFP) generally outperform the strong baseline and CRL on various metrics of failure prediction.

Algorithm 1: FMFP: Flat Minima for Failure Prediction algorithm

Input: Model Weights $\boldsymbol{\theta}$, scheduled learning α, cycle length c, number of iterations K, averaging start epoch S, neighborhood size ρ, the number of past checkpoints to be averaged n, loss function $\mathcal{L}$

Output: Model trained with FMFP

for $i \leftarrow$ *1 to* K **do**

  Sample a mini-batch data

  Compute gradient $\nabla\mathcal{L}(\boldsymbol{\theta})$ of the batch's training loss

  Compute worst-case perturbation $\hat{\epsilon} \leftarrow \rho \frac{\nabla\mathcal{L}(\boldsymbol{\theta})}{\|\nabla\mathcal{L}(\boldsymbol{\theta})\|_2}$

  Gradient update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\nabla\mathcal{L}(\boldsymbol{\theta} + \hat{\epsilon})$

  if $i \geq S$ *and* $mod(i, c) = 0$ **then**

    $\boldsymbol{\theta}^t_{sswa} \leftarrow \frac{\boldsymbol{\theta}_{sswa}{}^{t-1} \times n + \boldsymbol{\theta}^t}{n+1}$

  else

    i ++

Table 2. Confidence estimation on Tiny-ImageNet dataset. The means and standard deviations over three runs are reported. AURC and E-AURC values are multiplied by 10^3, and NLL are multiplied by 10 for clarity. Remaining values are percentages.

Network	Method	AURC (↓)	E-AURC (↓)	FPR-95% TPR(↓)	AUROC (↑)	AUPR-Success(↑)	AUPR-Error(↑)	ECE (↓)	NLL (↓)
ResNet-18	baseline [7]	124.54±0.97	53.13±0.12	62.45±0.68	86.49±0.11	92.54±0.04	75.10±0.57	10.13±0.42	15.76±0.14
	CRL [13]	118.05±1.88	49.55±0.75	**60.65±1.85**	86.62±0.37	93.10±0.09	**75.47±1.30**	7.56±0.64	14.69±0.09
	ours	**107.01±1.17**	**46.88±0.89**	62.35±1.38	**86.86±0.27**	**93.65±0.11**	73.14±0.75	**4.79±0.20**	**13.17±0.05**
ResNet-50	baseline [7]	119.01±3.19	53.05±1.04	63.56±0.20	86.22±0.20	92.66±0.18	73.74±0.61	10.14±0.10	15.41±0.26
	CRL [13]	110.80±2.33	48.27±1.21	62.01±0.35	87.02±0.18	93.38±0.19	74.15±0.26	7.35±0.29	14.23±0.19
	ours	**98.43±1.17**	**42.55±0.89**	**60.71±1.38**	**87.71±0.27**	**94.28±0.11**	**74.19±0.75**	**4.85±0.20**	**12.57±0.05**

Comparison with ConfidNet. ConfidNet [2,3] is a failure prediction method that relies on misclassified samples in training set. Therefore, it can not be used for models with high training accuracy. Therefore, we make comparison on VGG network [17] following the setting in [2]. As reported in Fig. 3, our method consistently outperforms ConfidNet under various metrics.

Selective risk-coverage curves results. Fig. 4 plots more risk-coverage curves. As can be seen, our method yields lower risk at a given coverage.

Relation with SWAG. SWA-Gaussion (SWAG) [12] is a bayesian inference technique, and has shown its effectiveness for confidence calibration. We mainly differ from SWAG in the following three aspects. (1) *Technique*: SWAG needs to sample many *e.g.*, 100 times to obtain bayesian inference uncertainty while ours do not need. (2) *Insight*: SWAG mainly leverage the weight average property of SWA for bayesian approximate. We are motivated by the connection between flat minima and confidence separability, thus other techniques like SAM can also be used. (3) *Problem setting*: SWAG focuses on calibration while we focus on failure prediction.

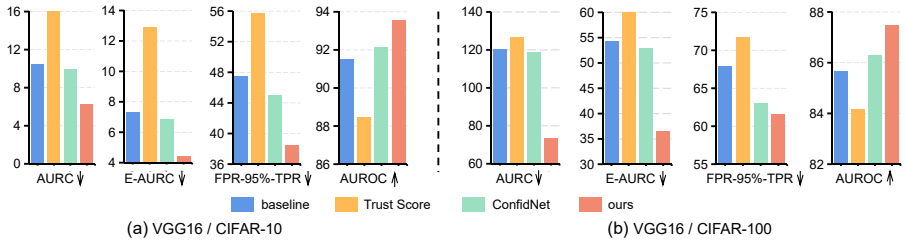

Fig. 3. Comparison with Trust Score and ConfidNet.

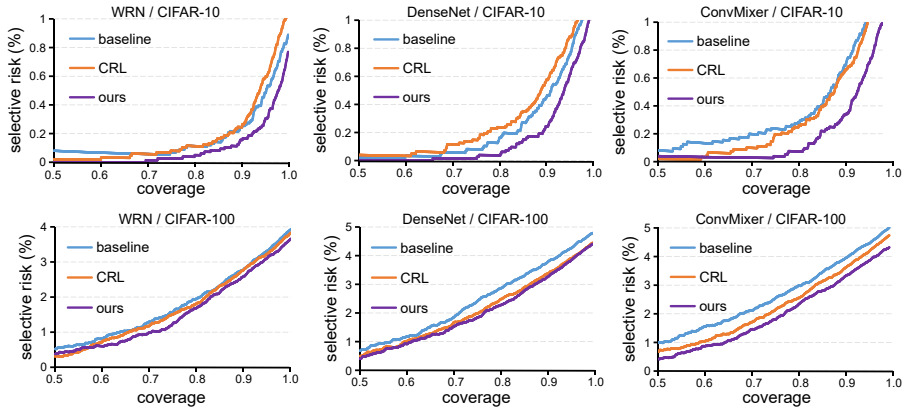

Fig. 4. Comparison of risk-coverage curves.

References

1. Brier, G.W.: Verification of forecasts expressed in terms of probability. Monthly Weather Review pp. 1–3 (1950) 2
2. Corbière, C., Thome, N., Bar-Hen, A., Cord, M., Pérez, P.: Addressing failure prediction by learning model confidence. In: NeurIPS. pp. 2898–2909 (2019) 6
3. Corbière, C., Thome, N., Saporta, A., Vu, T.H., Cord, M., Perez, P.: Confidence estimation via auxiliary models. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021) 6
4. Geifman, Y., El-Yaniv, R.: Selective classification for deep neural networks. In: NeurIPS. pp. 4878–4887 (2017) 1
5. Geifman, Y., Uziel, G., El-Yaniv, R.: Bias-reduced uncertainty estimation for deep neural classifiers. In: ICLR (2019) 1
6. Hastie, T.J., Tibshirani, R., Friedman, J.H.: The elements of statistical learning (2001) 2
7. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: ICLR (2017) 4, 6
8. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017) 4

9. Izmailov, P., Wilson, A., Podoprikhin, D., Vetrov, D., Garipov, T.: Averaging weights leads to wider optima and better generalization. In: UAI. pp. 876–885 (2018) 5
10. Joo, T., Chung, U.: Revisiting explicit regularization in neural networks for well-calibrated predictive uncertainty. arXiv preprint arXiv:2006.06399 (2020) 3, 4
11. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. IEEE Trans. Pattern Anal. Mach. Intell. pp. 318–327 (2020) 3
12. Maddox, W.J., Izmailov, P., Garipov, T., Vetrov, D.P., Wilson, A.G.: A simple baseline for bayesian uncertainty in deep learning. NeurIPS **32** (2019) 6
13. Moon, J., Kim, J., Shin, Y., Hwang, S.: Confidence-aware learning for deep neural networks. In: ICML. pp. 7034–7044 (2020) 5, 6
14. Mukhoti, J., Kulharia, V., Sanyal, A., Golodetz, S., Torr, P.H.S., Dokania, P.K.: Calibrating deep neural networks using focal loss. In: NeurIPS (2020) 3, 4
15. Müller, R., Kornblith, S., Hinton, G.: When does label smoothing help? In: NeurIPS. pp. 4696–4705 (2019) 3, 4
16. Naeini, M.P., Cooper, G.F., Hauskrecht, M.: Obtaining well calibrated probabilities using bayesian binning. In: AAAI. pp. 2901–2907 (2015) 2
17. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015) 6
18. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML. pp. 6105–6114 (2019) 4
19. Thulasidasan, S., Chennupati, G., Bilmes, J., Bhattacharya, T., Michalak, S.: On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In: NeurIPS. pp. 13888–13899 (2019) 2, 3, 4
20. Yao, L., Miller, J.: Tiny imagenet classification with convolutional neural networks. CS 231N 4, 5
21. Yun, S., Park, J., Lee, K., Shin, J.: Regularizing class-wise predictions via self-knowledge distillation. In: CVPR. pp. 13873–13882 (2020) 3, 4
22. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: Mixup: Beyond empirical risk minimization. In: ICLR (2018) 2
23. Zhang, W., Vaidya, I.: Mixup training leads to reduced overfitting and improved calibration for the transformer architecture. CoRR (2021) 3